<u>We Claim:</u>

1. A microprocessor die adapted for high-speed debugging comprising:

I/O pins on the die for making electrical connections between circuitry on the microprocessor die and external circuitry, the I/O pins including memory interface pins for connection to an external memory and debug interface pins for connection to an external in-circuit emulator (ICE);

a processor core for fetching and executing instructions;

a cache, coupled to the processor core, for supplying instructions and operands to the processor core;

a bus-interface unit, coupled to the cache and to the memory interface pins, for accessing the external memory when an instruction or an operand requested by the processor core is not present in the cache;

a debug queue, coupled to the processor core, for storing debug trace records generated by execution of traced instructions by the processor core; and

a debug interface, coupled to the debug queue and to the debug interface pins on the microprocessor die, for transferring debug trace records previously written to the debug queue to the external ICE, the external ICE for displaying the debug trace records;

wherein the debug interface pins are different pins than the memory interface pins, the debug interface being a separate interface from the memory interface,

whereby the debug queue buffers debug trace records to the external ICE using the debug interface pins and whereby bandwidth of the memory interface pins is not used for transferring debug trace records, allowing high-speed debugging.

2. The microprocessor die of claim 1 wherein the debug queue comprises a FIFO memory including:

writing means for writing the debug trace records to the debug queue at a first rate of a processor clock; and

reading means for reading the debug trace records stored in the debug queue to the debug interface pins at a second rate of an external clock,

wherein the second rate of the external clock is a lower rate than the first rate of the processor clock.

3. The microprocessor die of claim 2 wherein the reading means reads a debug trace record which was written to the debug queue at least N cycles of the external clock before, the debug queue containing N debug trace records,

whereby transfer of debug trace records to the external ICE is delayed by several external clock cycles when the debug queue contains other debug trace records.

4. The microprocessor die of claim 1 wherein the debug trace records stored in the debug queue include:

a time-stamp field for indicating a temporal location of when the debug trace record was generated by the processor core;

an identifier field for indicating a debug event which caused the debug trace record to be generated,

whereby the time-stamp field in the debug trace record is stored in the debug queue and transferred to the external ICE to indicate when the debug trace record was generated by the processor core.

5. The microprocessor die of claim 4 further comprising:

a time-stamp counter having a limited modulus, the time-stamp counter reaching the limited modulus in less than a minute when each pulse of a processor clock for clocking the processor core increments the time-stamp counter;

wherein the time-stamp field for a first debug trace record is capable of containing a same numerical value as a second debug trace record when the time-stamp counter reaches the limited modulus between the first debug trace record and the second debug trace record,

whereby the debug trace records include a time stamp generated from a limited-modulus counter.

6. The microprocessor die of claim 5 further comprising:
rollover means, coupled to the time-stamp counter, for writing a rollover trace record to the debug queue when the time-stamp counter reaches the limited modulus; and
reset means, coupled to the rollover means, for resetting the time-stamp counter when the time-stamp counter reaches the limited modulus,
whereby the rollover trace record in the debug queue separates the first debug trace record from the second debug trace record when the time-stamp counter reaches the limited modulus between the first debug trace record and the second debug trace record.

7. The microprocessor die of claim 5 further comprising:
divisor means, coupled to the time-stamp counter, for incrementing the time-stamp counter after every X pulses of the processor clock, where X is a clock divisor programmed into a clock divisor register,
whereby the time-stamp counter is incremented at a programmable rate.

8. The microprocessor die of claim 5 further comprising:
clearing means, coupled to the time-stamp counter, for clearing the time-stamp counter after each debug trace record is written to the debug queue,
wherein the time-stamp field indicates an amount of time since the previous debug trace record was written to the debug queue when the clearing means is activated.

9. The microprocessor die of claim 1 further comprising:
a second processor core for fetching and executing general-purpose instructions, the second processor core coupled to the cache and coupled to the debug queue;

the debug queue further coupled to the second processor core, for storing debug trace records generated by execution of traced instructions by the processor core and by the second processor core;

wherein the processor core and the second processor core are not directly connected to address and data I/O pins on the microprocessor die, the processor core and the second processor core indirectly accessing the external memory through the cache and the bus-interface unit,

whereby multi-processor debugging is accomplished by the debug queue buffering debug trace records generated from both the processor core and the second processor core.

10. The microprocessor die of claim 9 wherein the processor core and the second processor core execute independent programs.

11. The microprocessor die of claim 10 wherein the traced instructions are instructions which access a traced memory location, the traced memory location having a trigger address stored in a debug register, the microprocessor die further comprising:

trigger compare means, coupled to the processor core and coupled to the second processor core for comparing memory addresses generated by the processor core and the second processor core to the trigger address stored in the debug register, the trigger compare means signaling a debug event when a match is detected.

12. The microprocessor die of claim 1 further comprising:

a video controller for generating a horizontal synch signal and a vertical synch signal to an external display, the horizontal synch signal indicating when a new horizontal line of pixels is being sent to the external display, the vertical synch signal indicating when a new screen of horizontal lines is being sent to the external display;

28

pixel fetch means, in the video controller, for requesting pixels for display by the external display, the pixel fetch means requesting the pixels from the cache;

pixel transfer means, coupled to the pixel fetch means, for transferring the pixels from the cache to the debug queue,

whereby the debug queue stores the pixels for display.

13. The microprocessor die of claim 12 wherein the cache includes a frame buffer portion for storing a subset of the pixels in a screen of horizontal lines, the external memory storing a full frame buffer containing all of the pixels in the screen of horizontal lines,

wherein the pixel transfer means retrieves pixels from the frame buffer portion of the cache but retrieves pixels from the external memory only when the pixels are not present in the frame buffer portion of the cache,

whereby the frame buffer is cached.

14. The microprocessor die of claim 12 wherein the cache is a write-back cache, the cache containing updated pixels recently written by the processor core but not yet written back to the frame buffer in the external memory, the video controller first requesting pixels from the cache, the video controller requesting pixels from the frame buffer in the external memory when the pixels are not present in the cache,

whereby the frame buffer is cached by a write-back cache wherein updated pixels are not immediately written through to the frame buffer in external memory.

15. A microprocessor comprising:

a central processing unit (CPU) for executing instructions;

a cache for storing instructions, operands, and pixels for display, the cache operating in a write-back mode whereby an operand or a pixel written by the CPU is not written back to an external memory until an entire cache line containing the operand or the pixel is written back to the external memory;

a debug register for storing a trigger address of a debug event;

a debug comparator, coupled to the debug register and coupled to the CPU, for comparing the trigger address from the debug register to an address generated by execution of instructions by the CPU, the debug comparator signaling a debug event when a match occurs;

a trace record loader, coupled to the CPU, for generating a trace record when the debug event is triggered, the trace record including an identifier for the trigger address and a time stamp for indicating a relative time that the debug event occurred;

a FIFO buffer, coupled to the trace record loader, for storing the trace record for later transmission to an external in-circuit emulator (ICE); and

a video controller for transferring pixels from the cache to the FIFO buffer, the FIFO buffer transmitting the pixels to an external display when the external ICE is not connected to the microprocessor,

whereby the FIFO buffer stores trace records when debugging and pixels when not debugging.

16. The microprocessor of claim 15 further comprising:

a video-ICE interface, coupled to the FIFO buffer but not coupled to the cache, for transferring a stream of pixels from the FIFO buffer to the external display monitor when debug mode is disabled, but transferring trace records to the external ICE when debug mode is enabled;

a memory interface, coupled to the cache, for accessing an external DRAM memory when a request for an instruction, operand, or pixel misses in the cache,

wherein the video-ICE interface is separate from the memory interface so that trace records are not transferred over the memory interface, allowing the memory interface to operate at a full operating speed during debug mode.

17. The microprocessor of claim 16 wherein the CPU comprises a plurality of independent processor cores, each processor core for executing instructions

from a general-purpose instruction set independently of execution by other processor cores,

whereby debug events are generated from multiple processor cores executing independent programs.

18. A method for tracing execution of a program of instructions on a processor chip comprising the steps of :

executing a stream of instructions on a processor core and generating an address;

comparing the address generated by the processor core to a trigger address and trigger conditions and signaling a debug event when a match occurs;

reading an action field in a debug register containing the trigger address when the debug event is signaled and performing an action indicated by the action field;

when the action in the action field is a trace-address action:

generating a trace record including the address generated by the processor core and loading the trace record into a debug queue;

when the action in the action field is a trace-address-data action:

generating a trace record including the address generated by the processor core and data generated by the processor core and loading the trace record into a debug queue;

when one or more trace records are present in the debug queue:

reading an oldest trace record out of the debug queue and outputting the oldest trace record to debug interface pins on the processor chip

when a trace record is present on the debug interface pins:

transferring the trace record to an external in-circuit emulator (ICE) and displaying the trace record to a debugging user,

whereby trace records are generated and loaded into the debug queue before being transferred to the external ICE.

19. The method of claim 18 further comprising the steps of :

31

when the action in the action field is a trace-address-data action or a trace-address action:

reading a time-stamp counter and writing a time-stamp value of the time-stamp counter to the trace record generated

whereby the trace record includes the time-stamp value to indicate when the debug event occurred.

20. The method of claim 18 further comprising the steps of :

when the action in the action field is a stop-clock action:

stopping a processor clock to the processor core and halting execution of instructions;

when the action in the action field is a send interrupt action:

generating an interrupt to the processor core;

whereby the debug event generates a trace record, generates the interrupt or stops the processor clock.